So Wrong

Absurd ways of doing perfectly normal things

Why?

- We already know how to do these things
- By doing them in strange ways, maybe we can learn something new?

Web Frameworks

```
from flask import Flask
app = Flask(__name__)

@app.route("/")
def hello():
    return "Hello World!"
```

How about this instead?

```
from sowrong import GET, run


async def application():
    async for request, response in GET("/"):
        response.body = "hi"
```

Everything must eventually become SQL

```
>>> all_entries = Entry.objects.all()
```

```
>>> Blog.objects
<django.db.models.manager.Manager object at ...>
>>> b = Blog(name='Foo', tagline='Bar')
>>> b.objects
Traceback:
    ...
AttributeError: "Manager isn't accessible via Blog instances."
```

But could you write SQL without strings and ORMS?

```
def create_user_table():
    sql: create . table . user ( name . text, age . real)

def insert_values():
    sql: insert . into . user . values ('arjoonn', 600)

def show_table():
    sql: select * frm . user
```

Stalling APIs are bad. Offload to another process

```
from celery import Celery

app = Celery('tasks', broker='pyamqp://guest@localhost//')

@app.task
def add(x, y):
    return x + y
```

Or do we have to?

```
import bottle
from sowrong import batch


app = bottle.Bottle()


@batch(app.get("/do/<x>"))
def do(x):
    for i in range(int(x)):
        yield i
```

```
curl -L --max-redirs -1 http://localhost:8080/do/100000
```

So what do we learn?

Clear thought leads to exposed faults

```
from sowrong import GET, run


async def application():
    async for request, response in GET("/"):
        response.body = "hi"
```

Things that are forced, break down faster

```
def create_user_table():
    sql: create . table . user ( name . text, age . real)

def insert_values():
    sql: insert . into . user . values ('arjoonn', 600)

def show_table():
    sql: select * frm . user
```

A good solution solves all cases. Not special ones

```
import bottle
from sowrong import batch


app = bottle.Bottle()


@batch(app.get("/do/<x>"))
def do(x):
    for i in range(int(x)):
        yield i
```

Questions?

- Other things to check out
- https://github.com/satwikkansal/wtfpython
- http://entrian.com/goto/
- https://github.com/thesage21/SoWrong

- Arjoonn.com
- github.com/thesage21